Motion-Influenced Composition & *Transference*: Performance Experiments in Medium Mapping

Eli Stine
Oberlin College & Conservatory
Eli.Stine@oberlin.edu

ABSTRACT

Between 2009 and 2013 I have worked on a system in the MaxMSP, Jitter, and Processing environments that explores the transfer of data and gesture directly from movement to audio, audio to video, and movement to video in real-time. In this paper I describe the techniques used to capture the data, mapping schemes and their implications, the experience of performing the 2 incarnations of the piece at a variety of festivals and concerts, and possible future work.

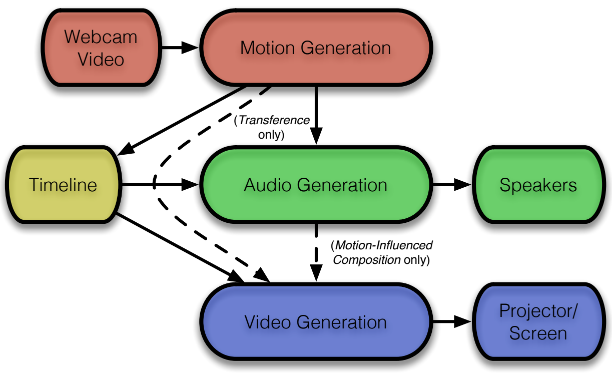
1. INTRODUCTION

Motion-Influenced Composition & *Transference* are two pieces spawned from my interest in how different mediums interact: audio representing movement, video representing audio, and video representing movement. These pieces depend on video input (specifically a webcam and body movement) and a system written in the MaxMSP [1], Jitter [2], and Processing [3] environments run on a computer that processes the motion input and generates audio and video in real-time.

The system is broken into 3 disjoint modules: motion generation, audio generation, and video generation. Diagram 1 illustrates the flow of data between the modules. Each module handles its input in different ways to produce meaningful information (movement, audio, and video data, respectively). The output of the audio and video generation modules constitutes the electronic component of the piece. Outside of these modules, a dynamic timeline (the pace of which is controlled by movement) modifies parameters (volume, brightness, distortion, for example) of the modules in real-time during performance.

A description of techniques employed in the different modules, the intra-systemic mappings of data used, experiential insights, and possible future work follows.

Diagram 1. Flow of Data between Modules in *Motion-Influenced Composition* & *Transference*



2. DATA GENERATION

2.1 Capturing Movement

2.1.1 Video Capture

Using Jitter, a low resolution (640 x 480) video is captured at 30 FPS and converted into luminance values. The luminance data is then processed above and below a threshold: pixels are made to be white if their luminance value is over a certain threshold and black if under that threshold. This binary pixel data is then analyzed using Open Source Computer Vision (OpenCV) [4] to get shape information: mass, orientation, perimeter, size, compactness, width-to-height ratio, density, and amount of movement in a direction, for example. The change of this shape information over time is the movement information data used by other modules in the system.

2.1.2 Video Normalization and Smoothing

In order for movement information to be captured as accurately as possible any brightness changes in the environment need to be minimized (the webcam is capturing light, and *not* movement data directly, after all). The gating threshold for the video is controlled during the performance in a self-reinforcing feedback loop to help reduce such environmental changes. A brightness level and sensitivity parameter are set by the performer when setting up the piece. The mean brightness of the current gated image is measured and if it falls outside of the brightness level plus or minus the sensitivity, the threshold is changed correspondingly in the opposite direction in order to normalize it.

2.1.3 Bounding Movement

Along with shape information a representation of the current overall amount of movement in the video input is measured. As each performance is in different conditions (lighting, distance from camera, etc.) high and low movement thresholds must be bounded to be universally usable. These bounds are set before performance. The program gathers the low movement threshold by measuring the absolute difference between consecutive frames within a window of 500 milliseconds while the performer moves as little as they will move during performance. The high threshold is found similarly by taking measurements while the performer moves as much as they will move during performance. The data within the two windows is then separately averaged and padded by a constant (to counteract over-sensitivity). The stream of movement information data within these bounds is referred to as the normalized absolute difference between consecutive frames (normabsdiff).

2.2 Generating Sound

2.2.1 Sound Techniques

The audio generation module in both *Motion-Influenced Composition* and *Transference* is itself modular: different sound generator submodules (MaxMSP encapsulations) are self-contained and triggered on and off by the timeline (see 4. Controlling Time for more info). A diversity of different techniques are used, ranging from frequency and ring modulation, equalization, convolution, distortion, granularization, and variable speed playback. Modules utilize recordings, generated noise, and simple waveforms.

2.2.2 Sound Content

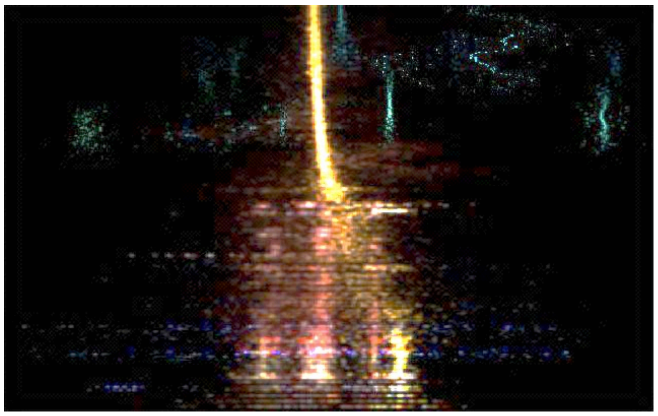
The more recent *Transference* utilizes more recordings (specifically solo voice and choral recordings) than *Motion-Influenced Composition*, which makes heavy use of synthesized sounds (simple waveforms, filtered noise, etc.) generated in real-time and then processed. This is a direct response to an insight into the relationship between control and identity when using recordings in such a context, which is discussed in more detail in subsection 3.1.4.

2.3 Generating Video in *Motion-Influenced Composition*

The video component of *Motion-Influenced Composition* is a visualization of an FFT of the summed stereo output of the audio generation module. The bins of the FFT are spread in the vertical dimension, stereo placement is calculated by comparing the levels of the individual bins in the 2 stereo channels and represented in the horizontal dimension, and the brightness and "noisiness" of the sound, calculated via Tristan Jehan's analyzer~ object [5], modifies color in the visualization. Parameters such as feedback, threshold and brightness of the individual RGB channels, and blurring can be set before or controlled during performance. A screenshot of video generated by the module can be seen in diagram 2.

A standalone visualizer, allowing for both live and sound file input, was released as the Sound Vision [6] application. This program was used in a performance of composer David Bird's piece for percussion and live electronics, "Vows", at Oberlin in 2010 [7].

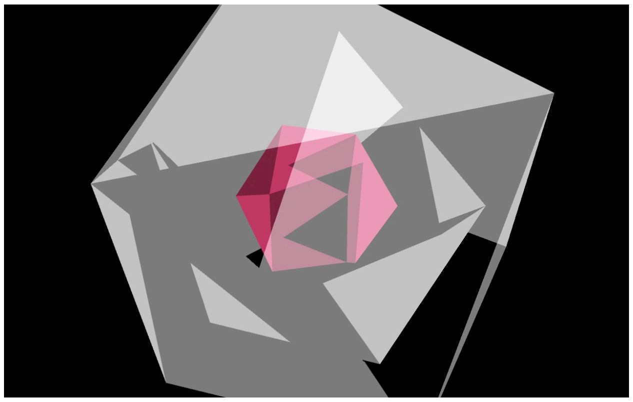
Diagram 2. Image capture of the climax of *Motion-Influenced Composition*



2.4 Generating Video in *Transference*

The video component of *Transference* utilizes the Processing programming language. 12 channels of Open Sound Control (OSC) [8] data are sent from MaxMSP to Processing via UDP. 3 different scenes in the Processing program are traversed in the piece, each of which is modified by some subset of the input parameters. OpenGL is used exclusively, contrasting greatly with the predominantly recording-based audio generated synchronously in a performance of the piece. The scenes vary from a slowly rotating, multi-faceted object illuminated by changes in movement directionality to a sea of shards that shimmer and spin in response to movement speed.

Diagram 3. Image capture of the second section of *Transference*



3. MEDIUM MAPPING

3.1 Audio Representing Movement

3.1.1 Historical Context

There exists a rich history of projects and instruments that create sound from movement information. Notable examples include the theremin [9], Michel Waisvisz' *The Hands* [10], Laetitia Sonami's *Lady's Glove* [11], Jeff Stolet's performances with proximity detectors [12], the Buchla Lightning [13], and Max Matthews' Radio-Baton [14]. These projects have at least two things in common: a performer interacts with some data-sensing device, and that device facilitates the generation of electronic sound or data. These pieces, including my own, extend the performer's body into the electronic domain. To enhance the instrumental aspect of *Motion-Influenced Composition* and *Transference* in most performances the computer is kept offstage, clearly establishing that the performer is interacting solely with the webcam as the music (and video) generating device.

3.1.2 Shape Information Routing

The individual parameters corresponding to shape data (collectively functioning as movement information) are routed to each of the audio generation submodules, either as an unmodified data stream or the difference between data points. Each submodule manipulates the input internally in a variety of ways depending on how sound is generated or triggered in that submodule. For example, a sound submodule that generates punctuating gestures may take the difference between size shape data points and calculate their average over some window. When the raw size shape data stream goes over the current average plus some constant offset an event is triggered, and a punctuating sound is heard that accompanies sudden or drastic changes in the perceived size of the video input.

3.1.3 Normalized Absolute Difference Between Consecutive Frames

Described in subsection 2.1.3 above, the normalized absolute difference between consecutive frames (normabsdiff) has a particularly prominent role in controlling and generating audio in both *Motion-Influenced Composition* and *Transference*. The reliability of this data is key for some submodules, which compare it to another movement information parameter in order to more accurately represent gestures. For example, a submodule may generate a sound if the density parameter changes relatively more than the normabsdiff in a given window.

3.1.4 Sonic Identity and Gestural Representation

Mapping movement information to synthesized sound and recorded material are very different processes. The sounds in *Transference* are almost exclusively recordings as opposed to *Motion-Influenced Composition*, which predominantly utilizes simple waveforms and shaped noise. This change in content was in response to an interest in exploring the importance of sonic identity in these pieces. I define sonic identity in this context as a sounds' capacity to convey exterior information (while still remaining linked to the gestures of the movement information).

There is a tradeoff between control and identity in mapping gestures to audio, whether it be recorded or synthesized material. At one side of this spectrum is triggered recording playback: a very simple reaction to a gesture, but having a strong, opaque identity, almost completely disconnected from the current context. At the other side is one or more synthesized sounds generated in real-time that have one or more features of the movement information mapped to them in some clearly audible way: every nuance of the gesture is mimicked, but there is a decided lack of identity. The sounds are completely transparent indicators of the current movement information.

The length of a sound is also important to how much identity it has. Assuming sounds are in response to movement information, as the time between responses to the movement information is reduced the ability of the audio to convey its own information is similarly reduced. For example, as the length of grains of a recording being produced by a granulator are reduced, their capacity to represent the gestures in the movement information at a smaller timescale increases, but their content is less immediately perceivable.

A mixture of different timescales of sounds, ranging from continuous beds of noise whose equalization is lightly modified by the movement information to extremely short, triggered sound gestures are used in these pieces to engage with ideas of control and identity. While this tradeoff comes up in other mappings, I feel it is particularly palpable between movement and its sonification.

3.2 Mapping To Video

3.2.1 Historical Context

Like audio representing movement, video representing media (particularly audio) has an interesting history. The first commercially available system to map audio to video was the Atari Video Music module [15], an entirely analog music visualizer released in 1976. This trend of commercial use seems to be much more prominent in the mapping of audio to video than in the aforementioned movement to audio mapping (excluding some commercial video games), and is exemplified by the Winamp [16] and iTunes [17] visualizers.

3.2.2 Audio to Video

The input to the video generation module in *Motion-Influenced Composition* is just the stereo output of the audio generation module. In creating submodules of the audio generation module this lack of dimensionality was taken into account: individual sounds were constructed and altered in response to *how they look when visualized* by the video generation module. Panning and spatialization were influenced by their graphical representation, and different timbres were chosen because of how they affect their corresponding color representation. A similar type of interaction via mapping occurs between the motion generation and video generation modules in the form of choreographical decisions influenced by the way in which they affect the video component.

3.2.3 Motion to Video

As mentioned earlier, the video generation module of *Transference* takes in 12 distinct parameters from the motion generation module. Note that this is a large dimensional expansion from the 2 channels of audio used to generate video in *Motion-Influenced Composition*, and that the data comes from the motion generation module rather than the audio generation module. This change significantly alters the system: rather than functioning in series the sound and video generation modules are working in parallel, representing the movement information in different mediums at roughly the same time.

The 12 parameters coming into the video generation module of *Transference* consist of streams of data that have a number of different roles: controlling the opacity of the sections (allowing for dynamic crossfades), conveying the directionality of the current movement information, expressing which samples are currently being triggered, and manipulating the speed, depth, or brightness of some element or process of the generated video. These data streams are derived from the timeline, OpenCV shape data, and the normabsdiff.

The mapping of motion to video is a curious mapping: both the input and output are in the same perceptual domain, vision. Because of this, the video functions more like an amplification of gestures rather than an extension (as was the case when mapping movement to audio). The perspicuity of the movement to video mapping is also different from cross-medium mappings: a one-to-one mapping is particularly clear and any complexity in the mapping process can be perceived as perplexing.

4. CONTROLLING TIME

4.1 System Design

Both *Motion-Influenced Composition* and *Transference* use a timeline that determines when, how, and with what intensity different sound submodules (and in *Transference*, video modules) operate. The timeline is advanced not by minutes or seconds but by gestural "happenings," small discrete units of movement. A "happening" occurs when the normabsdiff passes zero after rising and then falling, or vice versa. While somewhat complex, this measurement was created so that no small movements go unrepresented (resulting in unresponsiveness) and more expansive movements do not irrationally step forward the timeline (resulting in the piece being uncontrollable). With this in mind, the length of a performance of these pieces is *completely variable* and different textures and environments in the piece can be lengthened or shortened by modifying the amount of movement captured by the video generation module.

4.2 Types of Control

The most basic type of control imposed by the timeline is "on" and "off" sent to the audio generation submodules' volume controls or the video generation modules' opacity level. This may be preceded by a fade in or proceeded by a fade out. These fades in and out, or any other timeline actions for that matter, can be time-based (e.g. over 4 seconds) or occur over a certain number of gestural "happenings." At different points in the piece many aspects of the sound and video generation modules are altered, including the amount of distortion, the resonance of filter banks, the playback speed of grains in a granulator, or the rotation of an OpenGL object, for example. Timeline actions that are related are grouped into sections. Multiple submodules may be altered at the start of a section or a submodule may be altered after a delay at the end of the previous section, for example.

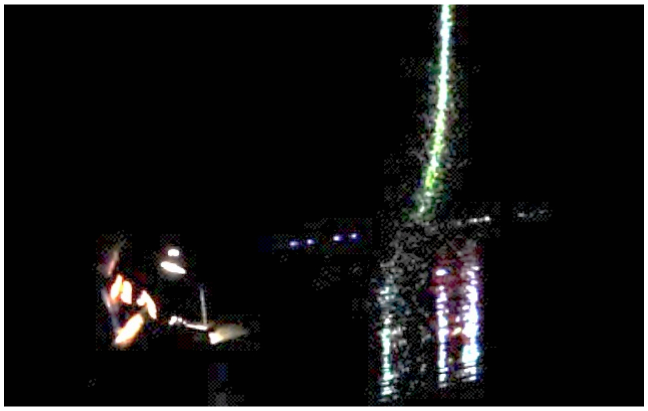
5. PERFORMANCE EXPERIENCE

Motion-Influenced Composition has been performed at the 2012 New Interfaces for Musical Expression (NIME) conference at the University of Michigan, at Oberlin Conservatory during my 2012 junior recital, and at the 2010 Threshold Festival at Bowling Green State University. *Transference* has been performed in 2013 at the Atlantic Center for the Arts in New Smyrna Beach, FL, where it was developed, and is programmed on my 2014 Oberlin Conservatory senior recital.

Before a performance of these pieces I practice playing the piece many times, choreographing a set of movements that is interesting, expressive, and allows for precise control. The movement information of the piece is primarily derived from my hand movements, but I am currently working with dancers to incorporate more of the body. Different sections require different levels of concentration and motor skill in order to shape and control the piece successfully as it develops. As the programmer of the system I may also make tweaks to the timeline and different modules during rehearsal, but try to keep this to a minimum, so that the gestures employed to give life to the piece remain a significant, integral part of its performance and are not just perceived as necessary input to a generative computer system doing all the work.

In order to get accurate movement information high contrast is needed between the performer and their surroundings. To aid in this, these pieces are performed in darkness with a single spot light on the performer. This frames the interaction between webcam and performer, highlighting the catalyst for all generative material in the piece.

Diagram 4. Performance of *Motion-Influenced Composition* at the 2010 Threshold Festival



6. FUTURE WORK

6.1 Alternative Motion Input

During the development of these pieces the Kinect controller [18] was released, and I experimented with it and the OpenKinect (then Freenect) [19] MaxMSP object. The added dimensionality, robustness, and precision of the Kinect could be advantageous to gaining more accurate, expressive control of the sound and video generation modules. Another "wires-free" alternative is the recently developed Leap Motion Controller [20]. The Playstation Move [21] and Wii [22] could also be used, although the ephemeral quality evoked by the performer not interfacing with hardware would be lost when using these systems.

6.2 The Inclusion of Other Mediums

While movement, generative music, and generative video art were the mediums that I explored with these pieces, the idea behind the system: modules related to a medium having a set of functions that are controlled by both a timeline and input from other modules, could be made to work with many other mediums. These include, but are not limited to: robotics, instrumental performance (with audio analysis in place of motion capture), tactile sensors, natural language generation, and speech analysis. Modifying the current systems of *Motion-Influenced Composition* and *Transference* with different mediums not only allows for the possibility of more fascinating medium mapping schemes, but could also suggest new and intriguing ways that the systems could be modified to incorporate these new mediums. The use of other mediums also suggests collaboration between artists and engineers other than myself, and could lend itself to ensemble and networked performance.

6.3 Performance Alternatives

6.3.1 Ensemble Performance

The incarnations of this piece thus far have only included a single performer (along with a single computer), but there is no technical reason why the piece could not be expanded to include multiple performers and/or computers. The members of a performing ensemble could either utilize independent systems (with their own input and output) or be part of an interconnected system that combines the data output from performers' generation modules into a collaborative, interactive amalgamation of their individual outputs.

6.3.2 Networked Performance

Because of the intermediary computer processing and discretization of all gestural and expressive aspects of the input to the systems used in these pieces, they are particularly applicable to networked performance. A performer of the piece could be a great distance away from the output of the piece, and furthermore, multiple performers in different locations could perform the piece collaboratively. The output of the different modules could be routed to any one or more locations, and could be modified along the way.

7. CONCLUSION

In this paper I discussed the technical details, mapping techniques used, and my experience performing the interactive multimedia pieces *Motion-Influenced Composition* and *Transference*. I also touched on how these projects relate to gesture, human-computer interaction and other works that map different mediums, and possible future expansions of these works. As the limitations of real-time analysis and video and audio generation dissolve I look forward to continuing to explore different ways to map them to other mediums, and developing the systems outlined in this paper in response to insights gained over the past 4 years working on them. Recent experiments involving collaboration with dancers have been particularly fruitful.

8. REFERENCES

[1] MaxMSP. Cycling '74, Walnut, CA, 2013; software available at http://cycling74.com/products/max/.

[2] Jitter. Cycling '74, Walnut, CA, 2013; software available at http://cycling74.com/products/max/.

[3] Processing. Processing Foundation, New York, NY, 2013; software available at https://processing.org/download/

[4] Open Source Computer Vision (OpenCV). itseez, Nizhny Novgorod, Russia; software available at http://opencv.org/downloads.html

[5] analyzer~. Tristan Jehan, 2010; software available at http://web.media.mit.edu/~tristan/maxmsp.html

[6] Sound Vision. Eli Stine, 2011; software available at http://www.oberlin.edu/student/estine/programs/Sound%20Vision.zip

[7] Bird, David, live electronics and Smith, Christian, percussion. "Vows"; viewable at http://vimeo.com/18852942

[8] Open Sound Control (OSC). CNMAT, Berkely, CA, 2002; software available at http://opensoundcontrol.org/introduction-osc

[9] Patented by inventor Léon Theremin in 1928. More information available at http://www.thereminworld.com/

[10] First version developed in 1984, more information available at http://crackle.org/

[11] First version developed in 1991, more information available at http://www.sonami.net/

[12] More information available at http://pages.uoregon.edu/fmo/stolet/

[13] More information available at http://buchla.com/

[14] More information available at http://www.csounds.com/mathews/

[15] More information available at http://www.atarimuseum.com/videogames/dedicated/videomusic/videomusic.html

[16] Winamp. Nullsoft, San Francisco, CA, 1997; software available at http://www.winamp.com/

[17] iTunes. Apple, Cupertino, CA, 2001; software available at http://www.apple.com/itunes/

[18] More information available at http://www.xbox.com/en-US/kinect

[19] libfreenect. Open Kinect, 2012; software available at https://github.com/OpenKinect/libfreenect

[20] More information available at https://www.leapmotion.com/

[21] More information available at http://us.playstation.com/ps3/playstation-move/

[22] More information available at http://wii.com//